### Sottoreti

La rete è composta da 4 sottoreti:

* **La prima sottorete include** host-A e router-1. La sottorete è un /23 e quindi tu puoi ottenere come risultato di indirizzi IP 232-23-2 = 510 differenti host. (495 minimum required)
* **La seconda sottorete include** host-B e router-1. La sottorete è un /24 e quindi tu puoi ottenere come risultato di indirizzi IP 232-24-2 = 254 differenti host. (140 minimum required)
* **La terza sottorete include** router-1 e router-2. La sottorete è un /30 e quindi tu puoi ottenere come risultato di indirizzi IP 232-30-2 = 2 differenti host.
* **La quarta sottorete include** router-2 e host-C. La sottorete è un /24 e quindi tu puoi ottenere come risultato di indirizzi IP 232-24-2 = 254 differenti host. (155 minimum required)

Vlan

Il router-1 si connette a 2 sottoreti diverse su una porta unica grazie a 2 diverse Vlan. Queste 2 Vlan cono contrassegnate da 2 divesi VIDs e sono 10 e 20.

### Interface-IP mapping

|  |  |  |  |
| --- | --- | --- | --- |
| Device | Interface | IP | Subnet |
| host-A | enp0s8 | 11.1.0.2/23 | Prima rete |
| router-1 | enp0s8.10 | 11.1.0.1/23 | Prima rete |
| host-B | enp0s8 | 12.0.1.2/24 | Seconda rete |
| router-1 | enp0s8.20 | 12.0.1.1/24 | Seconda rete |
| host-C | enp0s8 | 13.0.1.34/24 | Terza rete |
| router-2 | enp0s8 | 13.0.1.33/24 | Terza rete |
| router-1 | enp0s9 | 10.0.1.37/30 | Quarta rete |
| router-2 | enp0s9 | 10.0.1.38/30 | Quarta rete |

### Vagrant file and provisioning scripts

### Il progetto ha diversi file, ma il più importante è quello di Vagrantfile perché configura e gestisce i vari componenti della rete. Tutte le macchine virtuali sono basate su ‘ubuntu/bionic64’ perché è compatibile con tutte le cose che servono per il progetto.

### Router 1

### Il codice nel file Vagrantfile che appartiene a questo router è il seguente:

config.vm.define "router-1" do |router1|

    router1.vm.box = "ubuntu/bionic64"

    router1.vm.hostname = "router-1"

    router1.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_router-south-1", auto\_config: false

    router1.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_router-inter", auto\_config: false

    router1.vm.provision "shell", path: "router-1.sh"#, run: 'always'

    config.ssh.insert\_key = false

    router1.vm.provider "virtualbox" do |vb|

      vb.memory = 256

    end

Si vede chiaramente che si avvia lo script ‘router-1.sh’ .

Questo file è il seguente:

export DEBIAN\_FRONTEND=noninteractive

# Startup commands go here

apt-get update

apt-get install -y tcpdump curl traceroute --assume-yes

# Make sure the kernel forwards packets

sysctl net.ipv4.ip\_forward=1

# Set-up the interfaces

ip link add link enp0s8 name enp0s8.10 type vlan id 10

ip link add link enp0s8 name enp0s8.20 type vlan id 20

sudo ip link set dev enp0s8 up

sudo ip link set dev enp0s9 up

sudo ip link set dev enp0s8.10 up

sudo ip link set dev enp0s8.20 up

sudo ip addr add 11.1.0.1/23 dev enp0s8.10

sudo ip addr add 12.0.1.1/24 dev enp0s8.20

sudo ip addr add 10.0.1.37/30 dev enp0s9

#Routing rules

sudo ip route del default

sudo ip route add default via 10.0.1.38

Inizialmente si scaricano le librerie, successivamente si specificano gli indirizzi IP per le Vlan e interfacce, infine ci si occupa del routing

#### Router 2

### Il codice nel file Vagrantfile che appartiene a questo router è il seguente:

config.vm.define "router-2" do |router2|

    router2.vm.box = "ubuntu/bionic64"

    router2.vm.hostname = "router-2"

    router2.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_router-south-2", auto\_config: false

    router2.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_router-inter", auto\_config: false

    config.ssh.insert\_key = false

    router2.vm.provision "shell", path: "router-2.sh"#, run: 'always'

    router2.vm.provider "virtualbox" do |vb|

      vb.memory = 256

    end

Si vede chiaramente che si avvia lo script ‘router-2.sh’ .

Questo file è il seguente:

export DEBIAN\_FRONTEND=noninteractive

# Startup commands go here

apt-get update

apt-get install -y tcpdump curl traceroute --assume-yes

# Make sure the kernel forwards packets

sysctl net.ipv4.ip\_forward=1

# Set-up the interfaces

sudo ip link set dev enp0s8 up

sudo ip link set dev enp0s9 up

sudo ip addr add 13.0.1.33/24 dev enp0s8

sudo ip addr add 10.0.1.38/30 dev enp0s9

#Easy way: forward everything to the other router

sudo ip route del default

sudo ip route add default via 10.0.1.37

Prima si scaricano le librerie, poi si configurano gli indirizzi IP ed infine ci si occupa del routing.

Switch

### Il codice nel file Vagrantfile che appartiene allo switch è il seguente:

config.vm.define "switch" do |switch|

    switch.vm.box = "ubuntu/bionic64"

    switch.vm.hostname = "switch"

    switch.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_router-south-1", auto\_config: false

    switch.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_host\_a", auto\_config: false

    switch.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_host\_b", auto\_config: false

    config.ssh.insert\_key = false

    switch.vm.provision "shell", path: "switch.sh"#, run: 'always'

    switch.vm.provider "virtualbox" do |vb|

      vb.memory = 256

    end

Si vede chiaramente che si avvia lo script ‘switch.sh’ .

Questo file è il seguente:

export DEBIAN\_FRONTEND=noninteractive

apt-get update

apt-get install -y tcpdump

apt-get install -y openvswitch-common openvswitch-switch apt-transport-https ca-certificates curl software-properties-common

# Startup commands for switch go here

ovs-vsctl add-br switch

sudo ip link set dev enp0s8 up

sudo ip link set dev enp0s9 up

sudo ip link set dev enp0s10 up

# The access ports

ovs-vsctl --may-exist add-port switch enp0s9 tag=10

ovs-vsctl --may-exist add-port switch enp0s10 tag=20

# And the trunk link

ovs-vsctl --may-exist add-port switch enp0s8

Prima si scaricano le librerie e poi bisogna occuparsi di configurare un bridge e aggiungere le interfacce ad esso.

#### Host A e Host B

Il codice di questi 2 sono quasi uguali perché cambiano solo gli indirizzi IP .

### Il codice nel file Vagrantfile che appartengoono all’ host A e all’ host B sono i seguenti:

config.vm.define "host-a" do |hosta|

    hosta.vm.box = "ubuntu/bionic64"

    hosta.vm.hostname = "host-a"

    hosta.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_host\_a", auto\_config: false

    config.ssh.insert\_key = false

    hosta.vm.provision "shell", path: "host-a.sh"#, run: 'always'

    hosta.vm.provider "virtualbox" do |vb|

      vb.memory = 256

    end

  end

  config.vm.define "host-b" do |hostb|

    hostb.vm.box = "ubuntu/bionic64"

    hostb.vm.hostname = "host-b"

    hostb.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_host\_b", auto\_config: false

    hostb.vm.provision "shell", path: "host-b.sh"#, run: 'always'

    config.ssh.insert\_key = false

    hostb.vm.provider "virtualbox" do |vb|

      vb.memory = 256

    end

Come si può vedere i 2 codici sono molto simili e richiamano rispettivamente i propri ‘host-a.sh’ e ‘host-b.sh’ .

I 2 codici degli script sono i seguenti e sono molto simili:

--Host A

export DEBIAN\_FRONTEND=noninteractive

# Startup commands go here

apt-get update

apt-get install -y tcpdump curl traceroute --assume-yes

ip link set dev enp0s8 up

ip addr add 11.1.0.2/23 dev enp0s8

ip route del default

ip route add default via 11.1.0.1

Host B

export DEBIAN\_FRONTEND=noninteractive

# Startup commands go here

apt-get update

apt-get install -y tcpdump curl traceroute --assume-yes

ip link set dev enp0s8 up

ip addr add 12.0.1.2/24 dev enp0s8

ip route del default

ip route add default via 12.0.1.1

Per entrambi i casi devo configurare le interfacce, gli indirizzi ip e collegarli allo steesso router 1, infine avvio docker installando prima la sua libreria.

Host C

### Il codice nel file Vagrantfile che appartiene all’ host C è il seguente:

config.vm.define "host-c" do |hostc|

    hostc.vm.box = "ubuntu/bionic64"

    hostc.vm.hostname = "host-c"

    hostc.vm.network "private\_network", virtualbox\_\_intnet: "broadcast\_router-south-2", auto\_config: false

    config.ssh.insert\_key = false

    hostc.vm.provision "shell", path: "host-c.sh"#, run: 'always'

    hostc.vm.provider "virtualbox" do |vb|

    vb.memory = 512

    end

Si vede chiaramente che si avvia lo script ‘switch.sh’ , poi che la memoria data è 512 e non 256 perché ci dava un errore che diceva:memoria piena

Questo file è il seguente:

export DEBIAN\_FRONTEND=noninteractive

# Startup commands go here

sudo su

sudo apt-get update

sudo apt-get install -y tcpdump curl traceroute --assume-yes

sudo su

# Install docker and run nginx inside

sudo apt-get update

sudo apt-get install -y apt-transport-https ca-certificates curl software-properties-common --assume-yes --force-yes

sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add -

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

sudo apt-get update

sudo apt-get install -y docker-ce --assume-yes --force-yes

sudo mkdir -p /var/www

sudo chmod +r /var/www

sudo echo "Just a test page!" > /var/www/index.html

sudo docker run --name docker-nginx \

--restart=always \

-p 80:80 -d \

-v /var/www:/usr/share/nginx/html:ro \

nginx

ip link set dev enp0s8 up

ip addr add 13.0.1.34/24 dev enp0s8

ip route del default

ip route add default via 13.0.1.33

Prima si scaricano le librerie e poi bisogna creare una pagina index.html , ppoi si avvia docker, infine si configurano sia le interfacce sia il collegamento al router 2

## How-to

* Install Virtualbox
* Install Vagrant
* Clone this repository

~$ git clone <https://github.com/sixtynicooo/dncs-lab-funzionante-e-pronto.git>

Per partire il progetto bisogna usare il comando ‘vagrant up’, in alternativa si può usare il seguente comando per far eseguire il file per il test ed è ‘sh test.sh’ che controlla se i vari collegamenti tra gli host sono funzionanti.

Problemi

Il progetto è stato fatto usando window e abbiamo avuto alcuni problemi: il primo è che a volte mi dava un errore con permesso negato usando ‘vagrant ssh xxxx’ e per risolvere il problema ho usato la linea di codice ‘config.ssh.insert\_key = false’ nel file di Vagrantifile. Un altro problema era riservato agli host, sostanzialmente una parte dell’ errore era ‘ssh responded with non-zero status’ e per risolverlo si doveva configurare una variabile di vagrant usando il terminale di window e digitando il seguente codice ‘ SET VAGRANT\_PREFER\_SYSTEM\_BIN=0’.

Complessivamente è stato interessannte questo progetto perché fa vedere come collegare i vari ip e come interagiscono i vari componenti della rete.